(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 395 000 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
03.03.2004 Bulletin 2004/10

(51) Int Cl.[7]: **H04L 12/56**, H04L 29/08, H04L 29/06

(21) Application number: **02019051.8**

(22) Date of filing: **27.08.2002**

(84) Designated Contracting States:
AT BE BG CH CY CZ DE DK EE ES FI FR GB GR IE IT LI LU MC NL PT SE SK TR
Designated Extension States:
AL LT LV MK RO SI

(71) Applicant: **MATSUSHITA ELECTRIC INDUSTRIAL CO., LTD.**
Kadoma-city, Osaka 571-8501 (JP)

(72) Inventors:
- **Rey, José Luis**
  64287 Darmstadt (DE)
- **Burmeister, Carsten**
  22081 Hamburg (DE)
- **Hakenberg, Rolf**
  64295 Darmstadt (DE)

(74) Representative: **Grünecker, Kinkeldey, Stockmair & Schwanhäusser Anwaltssozietät**
Maximilianstrasse 58
80538 München (DE)

(54) A method of transmitting data streams dependent on the monitored state of the client application buffer

(57) A method of transmitting data streams between a sending host and a receiving host using a transport protocol, comprising the step of monitoring the occupancy state of an application layer buffer of the receiving host, and the step of signalling stream control data through the sending host depending on the monitored state of the client application layer buffer.

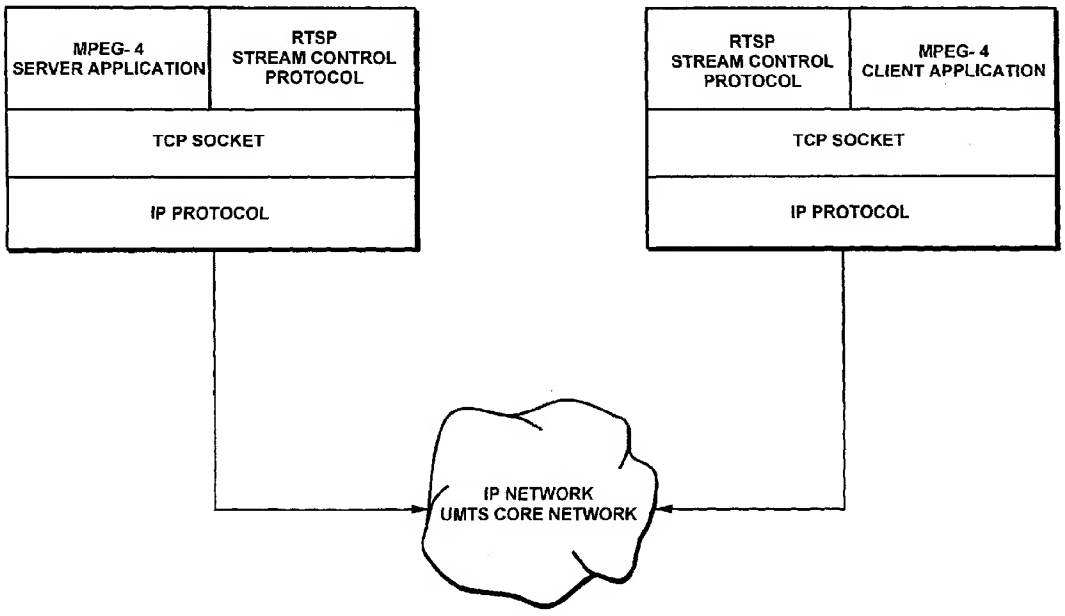

Fig. 1

EP 1 395 000 A1

Printed by Jouve, 75001 PARIS (FR)

Description

[0001] The present invention relates to a method of transmitting data streams between a sending host and a receiving host using a transport protocol such as TCP.

[0002] In network communications between a sending host and a receiving host, the transmission of data packets is mostly done on a best-effort basis. In the communication link, the available bandwidth and the current link conditions vary and hence, the end hosts must be provided with mechanisms to probe and adjust themselves to the current link conditions.

[0003] With the development of 3rd generation networks, so-called streaming applications are gaining importance. Streaming applications are understood as a continuous stream of data, e.g. video or audio data which is typically sensitive to time delays. In order that the user will be able to receive multimedia data streams on a portable receiver, for example a mobile phone, several requirements on the transport mechanism will be imposed in order that the delivered data meets with the resulting time constraints. These constraints are difficult to meet in a best-effort environment, which is subject to bandwidth variations and delays.

[0004] To cope with the problem of bandwidth adaption, it is common praxis to make several data streams with different encoding rates but same content available at the sending host. According to the available bandwidth, it is then switched between the data streams in order to adjust the rate to the available bandwidth. In other words, multi-rate content streams at the sending host are transmitted and switching operations between streams are performed depending on the available bandwidth.

[0005] There are some conventional approaches to TCP streaming that implement a client buffer management, but there are various constraints which are desired to overcome. First of all, the approach to implement a client buffer management should not require modifying the original encoded data or the TCP protocol in any way. Further, the need for additional application messages should not arise, otherwise, the method becomes difficult to implement. Finally, it should be avoided to discard data packets which leads to a loss of data and is therefore not an optimal solution.

[0006] In case of multi-media streaming using the TCP protocol, the specific problem lies in achieving a rate match between the bit rate which the TCP flow control mechanism is able to get out of the current transmission link and the rate at which the client application requests data. Assuming that multi-rate content is available at the server, the problem is reduced to how to make a decision to switch this stream and to which stream it should be switched.

[0007] Therefore, the problem underlying the present invention is to provide a method of transmitting data streams between a sending host and a receiving host, wherein said rate match is achieved in a simple manner and which is easy to implement.

[0008] This object is solved by a method comprising the steps as set forth in claim 1.

[0009] The idea underlying the invention is to monitor the occupancy state of the application layer buffer at the receiving host and to signal stream control data to the sending host depending on the monitored state of the application layer buffer. In this way, a close monitoring of the application layer buffer can be performed.

[0010] According to a preferred embodiment, the stream control data commands the sending host to limit or change the encoding rate of the data stream.

[0011] According to a further advantageous embodiment of the method, the stream control data signals to the sending host to switch from one data stream to another data stream having the same content but a different encoding rate. Consequently, the buffer monitoring operation triggers the issuing of appropriate requests to switch or to prepare the switch procedure between streams.

[0012] Preferably, the step of monitoring the occupancy state of the application layer buffer comprises to compare the bit rate at which the receiving host delivers new data to the application layer buffer with the rate at which data is requested by the application.

[0013] According to a further preferred embodiment, the actual occupancy state is compared against a plurality of occupancy thresholds, which is an easily implemented procedure. Upon exceeding or falling below certain buffer occupancy thresholds, appropriate action will be carried out to ensure a seamless stream switch.

[0014] The present invention will be more fully understood from the following detailed description with reference to the accompanying drawings which show:

Figure 1: the basic architecture of a sending host and a receiving host connected by a communications network in which the method subject to the present invention can be performed,

Figure 2: a block diagram of a server architecture as the sending host in which the method of the present invention is performed,

Figure 3: a block diagram of a client architecture, and

Figure 4: a schematic diagram of an application layer buffer illustrating several occupancy states.

[0015] Figure 1 illustrates the basic architecture of a communications network for connecting a video server as a sending host to a video client acting as a receiving host. The communications network is, for example, a UMTS core network, an external network, such as the internet or an internal intranet over which the server and

12/3/2008, EAST Version: 2.3.0.3

the client communicate by means of a transport protocol. As transport protocol TCP is selected as an example, but it is clear to a skilled person that any other transport protocol for transmitting data can be employed.

[0016] In a more general way, it will now be explained how TCP handles the exchange of data between the server and the client.

[0017] First of all, the video server needs to transfer its data to the protocol and tell it to send same to the client. To progressively read and send data, the protocol uses a kind of buffer with added functionalities, a so-called TCP socket. Generally, sockets are unequivocally identified by the IP addresses of the correspondents, the protocols and ports which are used. Once a connection between two sockets is established, the communication can be controlled by use of interface calls, which, for example may comprise "create a socket", "bind a socket to a transport address", "accept" or "close a connection", "send", "receive" or "write" data between the sender and the receiver. Wide-spread socket implementations are UNIX BSD 4.2 and Windows Winsock 2.0, where equivalent interface calls are implemented.

[0018] Each interface call has parameters, e.g. for the "send" call, parameters of relevance are the socket descriptor, the identifier of the data to be sent and the length of the data stream. Once a data stream is passed in one call, no single bit of this data stream can be modified. The only way to abort transfer of this data stream is to break down the connection.

[0019] Nevertheless, in order to close the connection, the application must explicitly issue a "close" call. Interface calls can be queued as, for example, in FTP. When asked for a file, the FTP server usually issues two interface calls for this transmission, namely a "send" call indicating the socket, the identifier of the data stream and the length thereof and a "close" call identifying the socket. The two interface calls are put into the queue and when the "send" call is completed, the connection is irreversibly broken down because the "close" call was next in the queue.

[0020] Alternatively, for example, in case of TELNET, the connection is established and the socket is set to wait for incoming data, i.e. commands typed on the keyboard.

[0021] Of course, the application can be configured such that the server only issues a send call and waits for the client to close. However, this is uncommon.

[0022] Additionally, it is assumed that the TCP protocol provides a means for asynchronously signaling certain information about events in the socket. Often in the specification the information will be an error message. In other cases, there will be information relation to the completion of processing a SEND or RECEIVE or another interface call.

[0023] Referring again to Figure 1, the video server provides, e.g. an MPEG 4 compressed data stream, i. e. a given video content, which is made available to the TCP socket in the form of mono-rate or multi-rate streams. Multi-rate streams are defined for the purpose of this and the following description, as one or several data streams with the same video content at different encoding rates. The MPEG data streams are forwarded to the TCP socket. Subsequently, by means of an IP protocol, data is transmitted to the video client. A stream control protocol, e.g. RTSP, should be used to perform such operations as PLAY, PAUSE or STOP the transmission.

[0024] With reference to Figure 2, which shows the basic architecture of a video server 200, at the MPEG server application 210, video content is made available in a buffer 211 at different coding rates. A switch 212, which receives stream control RTSP commands from RTSP block 220, such as "Play", "Pause", "Tear down" or "Change rate", selects one of the data streams which is forwarded to the TCP socket 230 in form of an MPEG file.

[0025] It is clear to a skilled person that other stream control protocols than RTSP could be used at the server and the client. The stream control protocol typically instructs to perform play, forwarding, rewinding, pause or stop functions. Additionally, if multi-rate content is available, switch to a higher or lower encoding rate can be instructed.

[0026] The server application forwards data segments of variable length to the TCP socket 230. The TCP socket forms packet data units (PDUs) for transmission to the client application.

[0027] TCP flow control continuously probes the network for available bandwidth by allowing one additional packet to be in flight every roundtrip time (RTT). Packets in flight are those which are sent without waiting for the respective acknowledgement. This continuous probing for bandwidth is unaware that the delivered bit rate of the TCP protocol might be too high for the receiving client application, i.e. available link bandwidth is 128 Kbps while the stream of data is 64 Kbps encoded video data. In this case, the application layer buffer would overflow and packet losses would take place. Similarly, the available bandwidth of the communication link might not be enough to transmit the data with the encoding rate that the client application would prefer. When such mismatches occur, packet losses take place. As a result, congestion control is invoked and the TCP throughput reduced. With the present invention, these rate mismatches are avoided.

[0028] The basic architecture of a TCP streaming client 300 is illustrated in Figure 3. The client 330 typically consists of a TCP socket 310, an application layer buffer 320, a buffer manager 330 for the application layer buffer and a display 350 for visualization purposes.

[0029] The client 300 processes the data segments of variable length as they come. In response thereto, it sends ACKs and performs flow control in order to prevent an overflow of the TCP socket 310 while extracting the maximum bit rate possible from the link. To do this, the TCP socket indicates to the server, with each ACK,

12/3/2008, EAST Version: 2.3.0.3

how much data it can receive until the socket is full using a window field in the header of the TCP ACK messages. If needed, the TCP client signals a zero-window telling the server to enter the "Persist" mode wherein no data is sent until the client allows to do so. Meanwhile the connection is maintained. The Persist mode and the signaling of a window field are standard functionalities of the TCP protocol. When the TCP socket has capacity for more data, it signals this to the server. Streaming control commands are signalled to the server by means of RTSP commands. In this manner, the frequency of sending rate reduction events can be decreased, and the delay and packet losses involved in abrupt connection teardowns are avoided.

[0030] The application layer buffer 320, also called "play-out buffer" receives data from the socket and transmits same to the MPEG client application 340. The manager 330 for the application layer buffer 320 receives a frame request from the MPEG client application and issues a socket request upon considering the state of the application layer buffer. The buffer manager 320 therefore insures that the bit rate with which the TCP socket 310 can deliver the new data to the application layer buffer 320 matches the bit rate at which the application layer buffer is emptied to the MPEG client application 340.

[0031] The buffer manager 330 is the one in charge of monitoring the fill-up status of the buffer. The TCP client socket 310 delivers its data to the buffer as long as there is capacity.

[0032] Let's now assume a loss-free (no bit-error losses) and finite bandwidth link. In the initial state the buffer occupancy has an optimum value, the buffer is in "OK" state, say between the maximum and minimum occupancy thresholds. At every given instant the manager 330 may observe one of the four different buffer states as illustrated in Figure 4.

- "Full": some maximum occupancy threshold (Max. Fill-up_Level + Receiver_Socket_Size <= Maximum Buffer Size) is surpassed or what's equivalent: the buffer manager is not able to accept the data from the TCP socket as fast as the socket is currently able to deliver, because the client application display rate is smaller than the available link bandwidth. This causes the TCP client to progressively decrease the advertised window until a zero-window is sent. Upon this, the server application enters the "Persist" Mode.

  In case multi-rate content is available at the server, the buffer manager additionally instructs the TCP server, via RTSP or another used stream control protocol, to send another stream with higher encoding rate. If not, the solution would be to limit the display rate, by just limiting the rate of the data that comes from the TCP socket.
- "Empty": some minimum occupancy threshold is not reached. This means that the buffer manager consumes the data in the buffer faster that the TCP socket can deliver it.

  If multi-rate content is available the solution is, via RTSP, to stop taking data from the current encoding rate and play from another stream encoded at a lower bit rate. If not, the solution is to stop the transmission, if the user desires.
- "Incipient Switch Downwards (Upwards)": these thresholds are placed between both "Full" and "Empty" thresholds. The fill-up rate of the buffer decreases (or increases), the buffer empties (or fills up) slowly. Action for the client to take: signal the server to initiate adequate actions to be prepared for a possible stream switch event.
- "OK": the buffer occupancy is optimum. It stays within the two "Incipient Switch Downwards (Upwards)" thresholds because both available bandwidth and display rate match .

[0033] This scheme can be easily generalized to the case of packet losses. The difference between "OK" and "Full (Empty)" fill-up levels shall account for TCP's instantaneous throughput variations. While the difference between the "Incipient Switch Downwards (Upwards)" and "Full (Empty)" fill-up levels shall avoid false switch decisions.

[0034] The above-described mechanism allows the client application to control and, if required, limit the sending rate of a server stream application. The protocol used by both applications is TCP, which includes flow and congestion control algorithms to avoid buffer overflow and is able to react appropriately in case of congestion. The described buffer monitoring operation is specifically useful when streaming real-time multimedia data over the TCP protocol and when the server is provided with multi-rate content that is the same multimedia content at different encoding rates. This scheme allows loss-free or near-loss-free switching between streams.

[0035] The above-described mechanism allows a client-driven sending rate limitation and/or rate adaption in case that multi-rate content is available to the server application.

Claims

1. A method of transmitting data streams between a sending host and a receiving host using a transport protocol, comprising the steps of:

   monitoring the occupancy state of a client application layer buffer of the receiving host, and

   signalling stream control data through the sending host depending on the monitored state of the application layer buffer.

2. The method according to claim 1, wherein the

12/3/2008, EAST Version: 2.3.0.3

stream control data commands the sending host to limit or change the encoding rate of the data stream.

3. The method according to claim 1 or 2, wherein the data stream is made available at the sending host in form of multi-rate streams of identical content.

4. The method according to claim 3, wherein the stream control data signals to the sending host to switch from one data stream to another data stream having the same content but a different encoding rate.

5. The method according to one of claims 1-4, wherein the step of monitoring the occupancy state of the application layer buffer comprises comparing the bit rate at which the receiving host delivers new data to the application layer buffer with the rate with which data is requested by the application.

6. The method according to one of claims 1-5, wherein the step of monitoring the occupancy state of the application layer buffer comprises the step of comparing the actual occupancy state against a plurality of occupancy thresholds.

7. The method according to one of claims 1-6, wherein the data streams are real-time multimedia data.

8. The method according to one of claims 1-7, wherein the transport protocol is TCP which includes a flow and congestion control algorithm which probes the communication link between the sending host and the receiving host for the available bandwidth.

9. The method according to one of claims 1-8, wherein the data streams are temporarily stored in an internal buffer at the receiving host.

10. The method according to one of claims 1-9, wherein the stream control data is in accordance with the RTSP protocol.

12/3/2008, EAST Version: 2.3.0.3

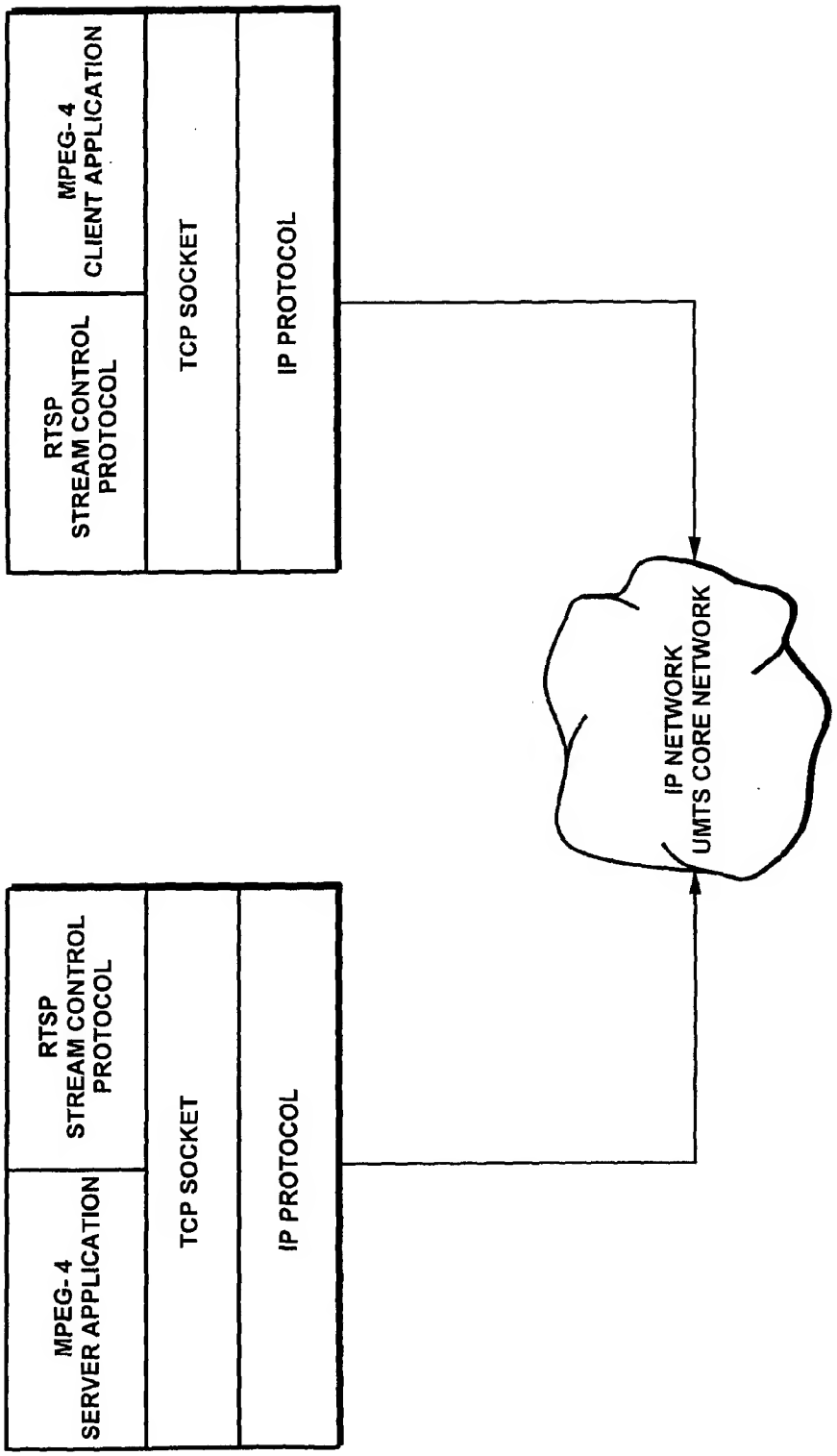
MPEG-4
CLIENT APPLICATION
RTSP
STREAM CONTROL
PROTOCOL
TCP SOCKET
IP PROTOCOL
IP NETWORK
UMTS CORE NETWORK
MPEG-4
SERVER APPLICATION
RTSP
STREAM CONTROL
PROTOCOL
TCP SOCKET
IP PROTOCOL

Fig. 1

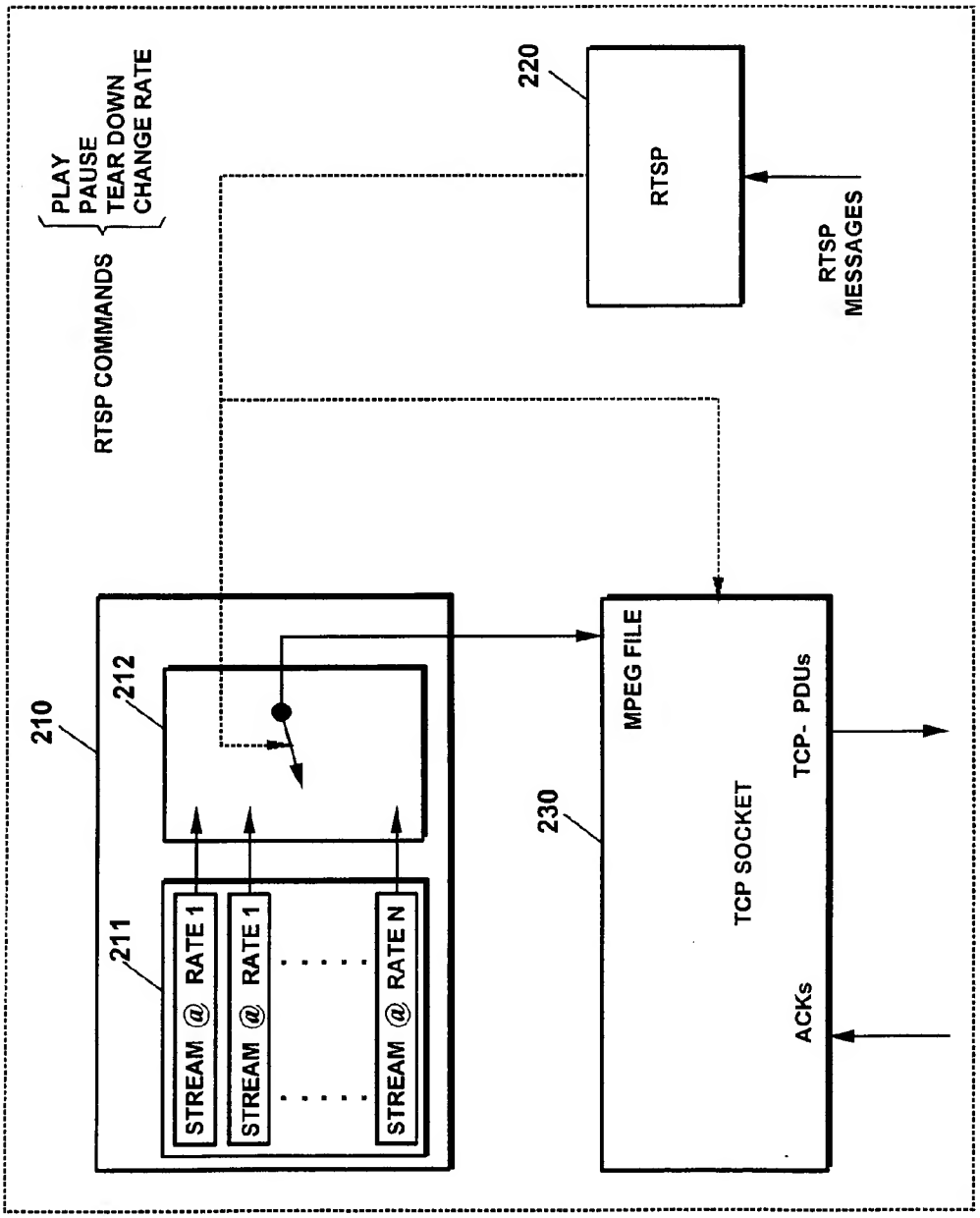

Fig. 2

12/3/2008, EAST Version: 2.3.0.3

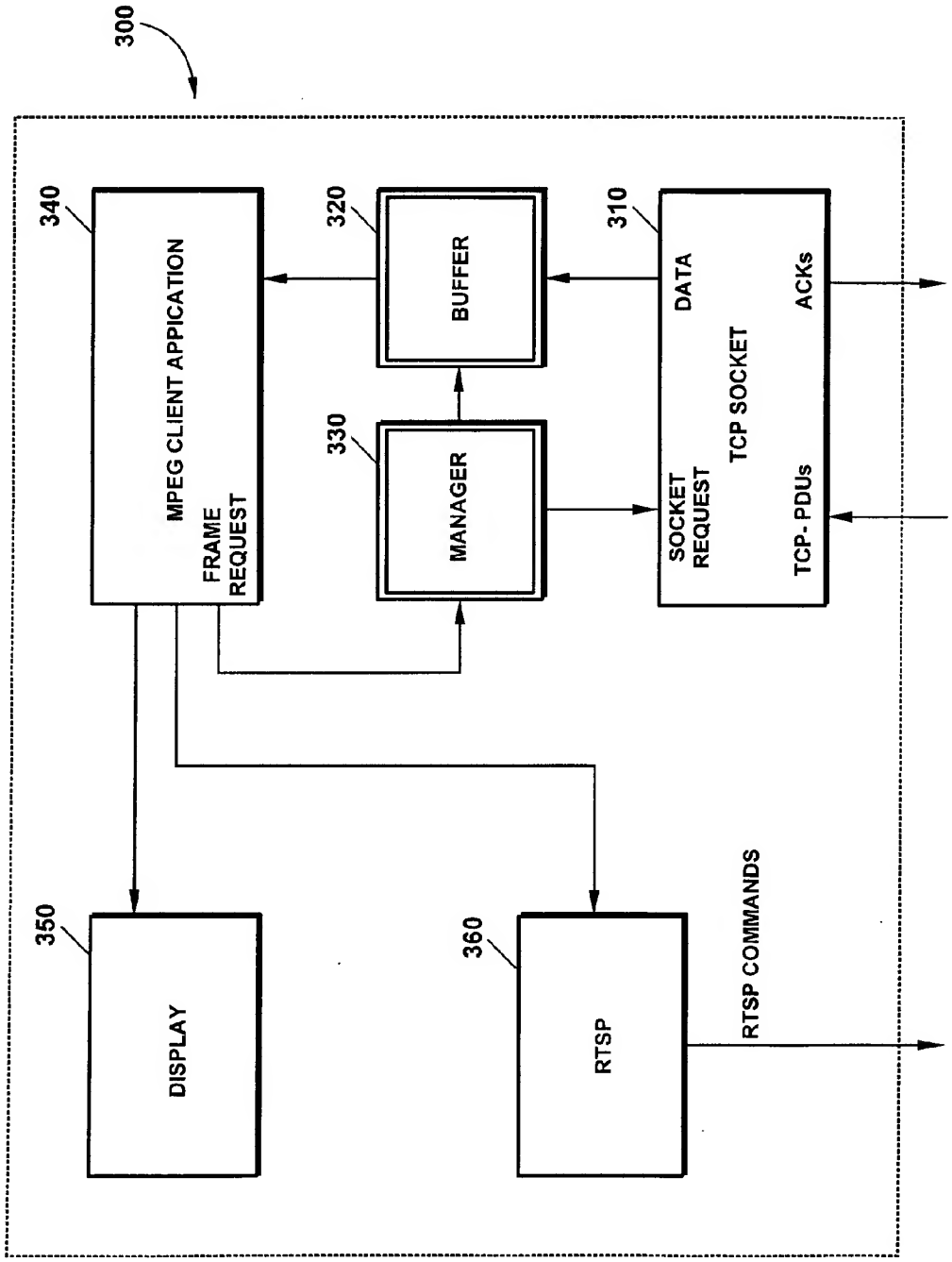

Fig. 3

12/3/2008, EAST Version: 2.3.0.3

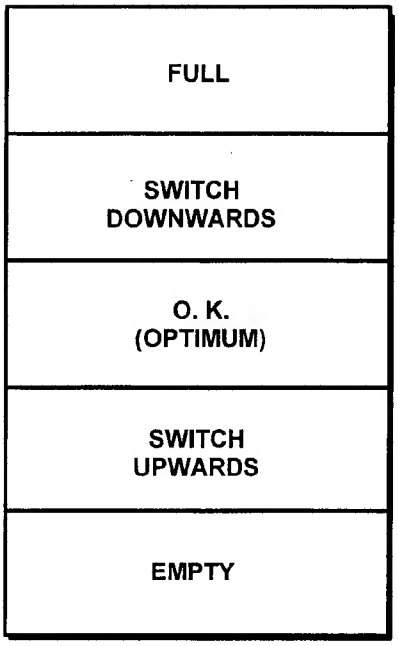

Fig. 4

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 02 01 9051

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
X	MIELKE M ET AL: "A multi-level buffering and feedback scheme for distributed multimedia presentation systems" COMPUTER COMMUNICATIONS AND NETWORKS, 1998. PROCEEDINGS. 7TH INTERNATIONAL CONFERENCE ON LAFAYETTE, LA, USA 12-15 OCT. 1998, LOS ALAMITOS, CA, USA,IEEE COMPUT. SOC, US, 12 October 1998 (1998-10-12), pages 219-226, XP010587051 ISBN: 0-8186-9014-3	1,2,5-9	H04L12/56 H04L29/08 H04L29/06
Y	* figures 1-3; table 1 * * page 219, paragraph 1.1 * * page 219, paragraph 1.2 - page 220 * * page 220, paragraph 2 * * page 220, paragraph 3.1 * * page 221, paragraph 3.2 - page 222 * * page 223, paragraph 4.2 - page 224 * ---	3,4,10	
Y	CONKLIN G J ET AL: "VIDEO CODING FOR STREAMING MEDIA DELIVERY ON THE INTERNET" IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, IEEE INC. NEW YORK, US, vol. 11, no. 3, March 2001 (2001-03), pages 269-281, XP001093477 ISSN: 1051-8215 * page 271, left-hand column, paragraph D; figures 1,2 * * page 271, left-hand column, paragraph E - right-hand column * * page 272, right-hand column, paragraph A - left-hand column * --- -/--	3,4,10	TECHNICAL FIELDS SEARCHED (Int.Cl.7) H04L

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
MUNICH	19 December 2002	Michael, T

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

12/3/2008, EAST Version: 2.3.0.3

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 02 01 9051

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
A	GRUBER S ET AL: "Protocol considerations for a prefix-caching proxy for multimedia streams" COMPUTER NETWORKS, ELSEVIER SCIENCE PUBLISHERS B.V., AMSTERDAM, NL, vol. 33, no. 1-6, June 2000 (2000-06), pages 657-668, XP004304799 ISSN: 1389-1286 * page 658, right-hand column, line 33, paragraph 45 * * page 659, left-hand column, line 42, paragraph 45 * ---		
A	HUI J Y ET AL: "CLIENT-SERVER SYNCHRONIZATION AND BUFFERING FOR VARIABLE RATE MULTIMEDIA RETRIEVALS" IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, IEEE INC. NEW YORK, US, vol. 14, no. 1, 1996, pages 226-237, XP000548824 ISSN: 0733-8716 * page 228, left-hand column, line 15 - line 34; figure 2 * * page 228, right-hand column, line 14 - line 33 * -----		TECHNICAL FIELDS SEARCHED (Int.Cl.7)

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
MUNICH	19 December 2002	Michael, T

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

12/3/2008, EAST Version: 2.3.0.3